Datos aleatorios, Regre.Lineal

September 17, 2020

1 Modelo de Regresión Lineal

1.1 Modelo con datos simulados

- y = a + b * x
- X = 100 valores distribuidos según una N(1.5,2.5), media y desviacion estandar
- Ye = 5 + 1 * X + e
- e estara distribuida segun una N(0,0.8), si media 0 no, ocurren desplazamientos en el modelo.

[103]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

[104]:
```
x = 1.5 + 2.5 * np.random.randn(100)
```

[105]:
```
res = 0 + 0.8 * np.random.randn(100)
```

[106]:
```
y_pred = 5 + 1 * x
```

[107]:
```
y_act = 5 + 1 * x + res
```

[108]:
```
x_list = x.tolist()
y_pred_list = y_pred.tolist()
y_act_list = y_act.tolist()
```

[109]:
```
data = pd.DataFrame(
{
    "x":x_list,
    "y_actual":y_act_list,
    "y_prediccion":y_pred_list
})
```

[110]:
```
data.head()
```

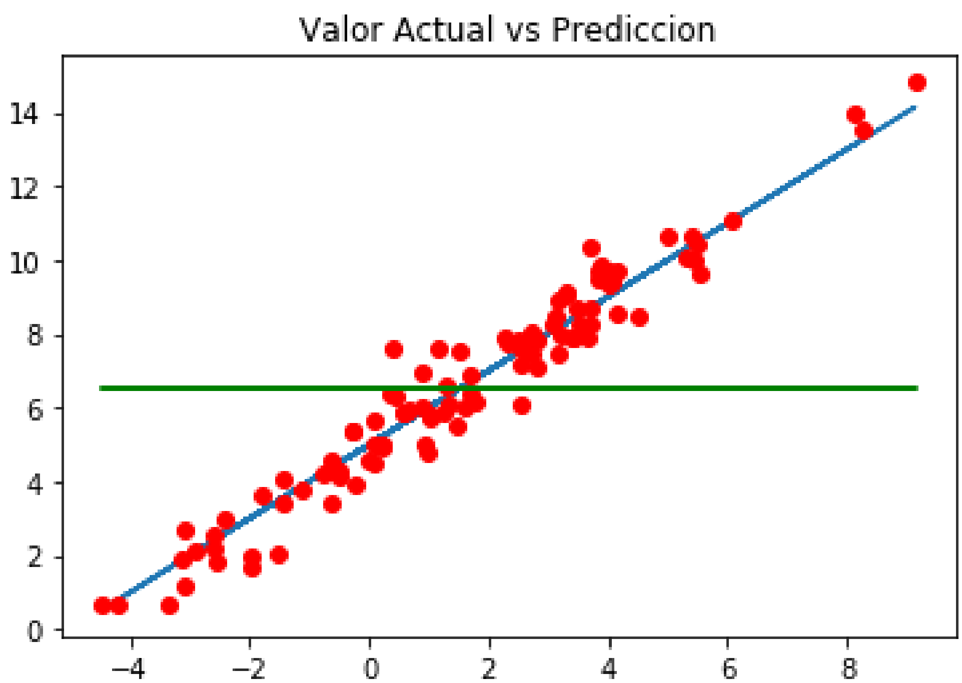
[110]:

	x	y_actual	y_prediccion
0	1.016354	5.701187	6.016354
1	2.514536	7.816246	7.514536
2	2.537257	6.124276	7.537257
3	-0.282380	5.335924	4.717620
4	8.254630	13.511166	13.254630

[111]:
```
y_mean =[np.mean(y_act) for i in range(1,len(x_list) + 1)]
```

[112]:
```
plt.plot(x,y_pred)
plt.plot(x,y_act,"ro")
plt.plot(x,y_mean,"g")
plt.title("Valor Actual vs Prediccion")
```

[112]: Text(0.5, 1.0, 'Valor Actual vs Prediccion')



[113]:
```
data["SSR"] = (data["y_prediccion"]-np.mean(y_act))**2
data["SSD"] = (data["y_prediccion"]-data["y_actual"])**2
data["SST"] = (data["y_actual"]-np.mean(y_act))**2
```

[114]:
```
data.head()
```

[114]:

	x	y_actual	y_prediccion	SSR	SSD	SST
0	1.016354	5.701187	6.016354	0.285815	0.099330	0.722133

```
1  2.514536   7.816246      7.514536   0.928458  0.091029   1.600923
2  2.537257   6.124276      7.537257   0.972761  1.996517   0.182069
3 -0.282380   5.335924      4.717620   3.361175  0.382300   1.476338
4  8.254630  13.511166     13.254630  44.939046  0.065811  48.444325
```

[115]:
```
SSR = sum(data["SSR"])
SSD = sum(data["SSD"])
SST = sum(data["SST"])
```

[116]:
```
SST
```

[116]: 854.4963653885451

[117]:
```
SSR
```

[117]: 749.3628729511602

[118]:
```
SSD
```

[118]: 44.79580780092574

[119]:
```
SSD + SSR
```

[119]: 794.158680752086

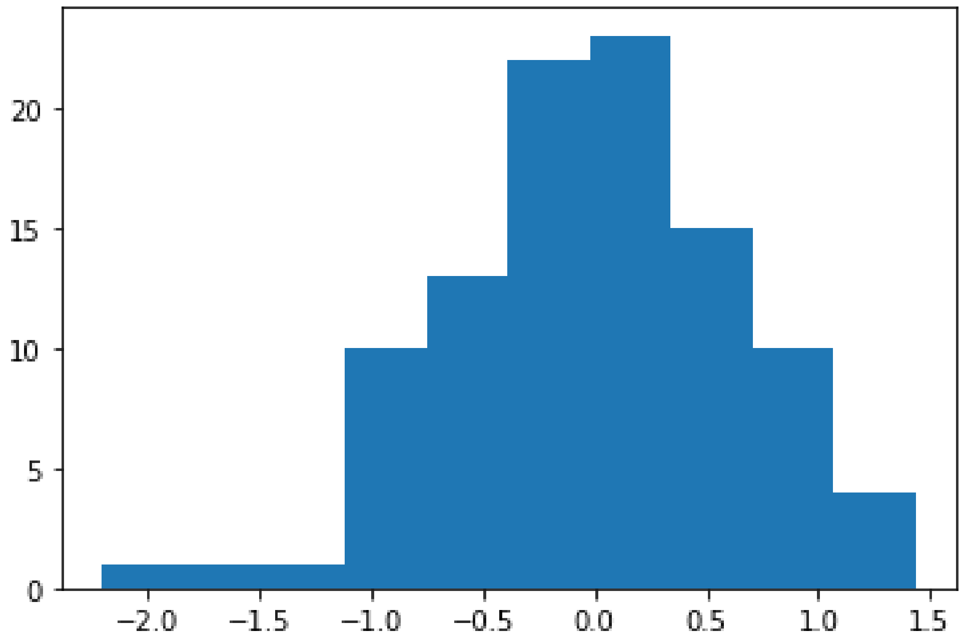
[120]:
```
R2 = SSR/SST
```

[121]:
```
R2
```

[121]: 0.8769643772684977

[123]:
```
plt.hist((data["y_prediccion"]-data["y_actual"]))
```

[123]:
```
(array([ 1.,  1.,  1., 10., 13., 22., 23., 15., 10.,  4.]),
 array([-2.20635247, -1.84242826, -1.47850405, -1.11457985, -0.75065564,
        -0.38673143, -0.02280723,  0.34111698,  0.70504119,  1.06896539,
         1.4328896 ]),
 <a list of 10 Patch objects>)
```



2 Obteniendo la recta de regresión

- y = a + b * x
- b = sum((xi - x_m)*(y_i-y_m))/sum((xi-x_m)^2)
- a = y_m - b * x_m

[124]:
```
x_m = np.mean(data["x"])
y_m = np.mean(data["y_actual"])
x_m, y_m
```

[124]:
```
(1.538192936084344, 6.550970491703897)
```

[126]:
```
data["beta_n"] = (data["x"]-x_m)*(data["y_actual"]-y_m)
data["beta_d"] = (data["x"]-x_m)**2
```

[127]:
```
beta = sum(data["beta_n"])/sum(data["beta_d"])
```

[128]:
```
alpha = y_m -beta *x_m
```

[129]:
```
alpha, beta
```

[129]:
```
(4.950816085564027, 1.040281988430695)
```

- El modelo lineal obtenido por regresion es: y = 4.950816085564027 + 1.040281988430695 * x

[130]:
```
data["y_model"] = alpha + beta * data["x"]
```

[131]:
```
data.head()
```

[131]:

	x	y_actual	y_prediccion	SSR	SSD	SST	\
0	1.016354	5.701187	6.016354	0.285815	0.099330	0.722133	
1	2.514536	7.816246	7.514536	0.928458	0.091029	1.600923	
2	2.537257	6.124276	7.537257	0.972761	1.996517	0.182069	
3	-0.282380	5.335924	4.717620	3.361175	0.382300	1.476338	
4	8.254630	13.511166	13.254630	44.939046	0.065811	48.444325	

	beta_n	beta_d	y_model
0	0.443450	0.272316	6.008111
1	1.235343	0.953246	7.566643
2	-0.426296	0.998129	7.590279
3	2.212081	3.314487	4.657061
4	46.747715	45.110522	13.537959

[133]:
```
SSR = sum((data["y_model"]-y_m)**2)
SSD = sum((data["y_model"]-data["y_actual"])**2)
SST = sum((data["y_actual"]-y_m)**2)
```

[134]:
```
SSR,SSD,SST
```

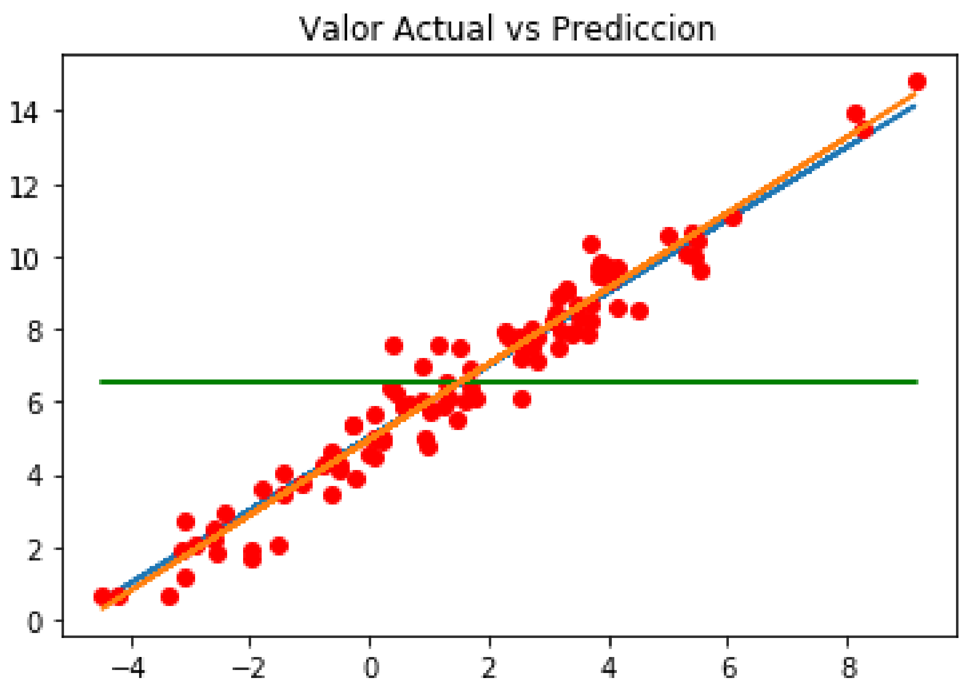
[134]: (810.9328028052313, 43.56356258331354, 854.4963653885453)

[135]:
```
R2 = SSR/SST
R2
```

[135]: 0.949018434310712

[141]:
```
y_mean =[np.mean(y_act) for i in range(1,len(x_list) + 1)]
plt.plot(x,y_pred)
plt.plot(x,y_act,"ro")
plt.plot(x,y_mean,"g")
plt.plot(data["x"],data["y_model"])
plt.title("Valor Actual vs Prediccion")
```

[141]: Text(0.5, 1.0, 'Valor Actual vs Prediccion')



2.1 Error estandar de los residuos (RSE)

[142]:
```
RSE = np.sqrt(SSD)/(len(data)-2)
```

[143]:
```
RSE
```

[143]: 0.06734969272609344

[144]:
```
np.mean(data["y_actual"])
```

[144]: 6.550970491703897

[145]:
```
RSE / np.mean(data["y_actual"])
```

[145]: 0.010280872553369706